UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/762,944	01/22/2004	Alexander J. Somogyi	BEAS-01337US2	5981

23910 7590 12/27/2007

FLIESLER MEYER LLP
650 CALIFORNIA STREET
14TH FLOOR
SAN FRANCISCO, CA 94108

EXAMINER	
DAVE, JYOTI D	

ART UNIT	PAPER NUMBER
4182	

MAIL DATE	DELIVERY MODE
12/27/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/762,944	SOMOGYI ET AL.	
	Examiner	Art Unit	
	JYOTI D. DAVE	4182	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on _____.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-10* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-10* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *22 January 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *4/21/2006*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-4 are rejected by 35 U.S.C. 103(a) as being unpatentable over Lampson et al. (5,335,343) in view of Nageswaran (5,991,792).

3. Lampson discloses a method for implementing a two-phase commit protocol, comprising: dispatching a first two-phase commit protocol operation from a first thread to a second thread, the first two phase commit protocol operation associated with a first resource and a first phase of two-phase commit protocol (see fig 4, 5, 6 and column 6, lines 45-50, in which Lampson discloses item sends "commit" message to all of its subordinates). Lampson is dispatching a phase 2 commit protocol operation of a two-phase commit protocol from the commit coordinator to the subordinates. One of ordinary skill in the art can appreciate that the subordinate could be considered a thread. The commit coordinator keeps track of the responses from the subordinates of the first phase (prepare to commit phase) of the 2 phase commit protocol. Only if the subordinate votes yes to commit during the first phase, does the second commit phase begin. Therefore, the commit coordinator has association with the subordinate. Also, the first phase effects the second phase. So, there is association between the first and the second phase of the 2 phase commit protocol.

4. Lampson also discloses processing a second two-phase commit protocol operation by the first thread, the second two-phase commit protocol operation associated with a second resource and the first phase of the two phase commit protocol (see fig 4, 5, 6 and column 6, lines 45-50, in which Lampson discloses item sends "commit" message to all of its subordinates). Lampson discloses having multiple threads. Therefore, it would be obvious to one of ordinary skill in the art that there will be a second two phase commit protocol operation in another thread. Lampson is dispatching a phase 2 commit protocol operation of a two-phase commit protocol from the commit coordinator to the subordinates. One of ordinary skill in the art can appreciate that the subordinate could be considered a thread. The commit coordinator keeps track of the responses from the subordinates of the first phase (prepare to commit phase) of the 2 phase commit protocol. The commit coordinator is in charge for the first phase of the multiple subordinates that will carry out the second phase of the two phase commit protocol. Only if the subordinate votes yes to commit during the first phase, does the second commit phase begin. Therefore, the commit coordinator has association with the subordinate. Also, the first phase effects the second phase. So, there is association between the first and the second phase of the 2 phase commit protocol.

5. Furthermore, Lampson discloses determining the first two-phase commit protocol operation is complete (see column 6 lines 55-65, in which Lampson discloses each subordinate receiving a commit message moves to a committing state, does a forced write of a commit record, and sends an "acknowledge" message to the coordinator.) The commit coordinator knows that once the acknowledgement has come back that the second phase commit is complete.

6. In addition, Nageswaran discloses managing a thread pool of reusable threads in a computer system. It would be obvious to one skilled in the art to have used reusable threads in conjunction with the 2-phase commit process described in Lampson. The motivation to use reusable thread would be they are available when a commit protocol is needed and are removed when not needed. This removal helps shrink the thread pool which is more efficient. Resources are no longer being used to keep non-used threads.

7. In reference to claim 2, claim 2 depends upon claim1. Claim 1 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792).

8. Nageswaran discloses selecting an idle server thread to process the first two-phase commit protocol operation (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, but also to distribute the threads when needed by the application. It would also be obvious to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

9. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit message to a subordinate (thread). Once the subordinate has the message it must commit.

If the thread is a reusable thread, as described in Nageswaran, the system must look for a thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

10. In reference to claim 3, claim 3 depends upon claim2. Claim 1 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792).

11. Nageswaran discloses determining available server threads in a server (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, but also to distribute the threads when needed by the application. It would also be obvious to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

12. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit message to a subordinate (thread). Once the subordinate has the message it must commit.

If the thread is a reusable thread, as described in Nageswaran, the system must look for a thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

13. In reference to claim 4, claim 4 depends upon claim3. Claim 3 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792).

14. Nageswaran discloses a thread pool manager determines the available server threads (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads, the available server threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, but also to distribute the threads when needed by the application. It would also be obvious to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

15. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit

message to a subordinate (thread). Once the subordinate has the message it must commit. If the thread is a reusable thread, as described in Nageswaran, the system must look for a thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

16. Claims 5-10 are rejected by 35 U.S.C. 103(a) as being unpatentable over Lampson et al. (5,335,343) in view of Nageswaran (5,991,792) and in further view of Haritsa (The PROMPT Real Time Commit Protocol, Feb. 2002).

17. Claim 5 is depends upon claim 1. Claim 1 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792). Lampson in view of Nageswaran does not disclose reporting results of the first and second two-phase commit protocol to a log. However, Haritsa discloses a 2 phase commit protocol where after the second phase of the 2 phase commit protocol, the results of the first and second 2 phase commit protocol are reported to a log (see pg. 162, section 3.1, third paragraph, in which Haritsa discloses each cohort, upon receiving the COMMIT message, moves to the committing state, force-writes a commit log record, and sends an ACK message to the master).

18. The motivation to combine the 2 phase commit protocol with reusable threads, as disclosed by Lampson in view of Nageswaran, with a commit log record is the commit log record will be able to keep records of what protocol has been committed and what has

been aborted. This information can be valuable to the IT specialists caring for the system or running the two phase commit protocol.

19. In reference to claim 6, Haritsa discloses a method for processing N two-phase commit protocol operations, comprising: processing N two-phase commit protocol operations in a first server thread, wherein the process for each of N-1 of the two-phase commit protocol operations (see pg. 162, section 3, paragraph 1, in which Haritsa discloses in this model, there is one process, called the master which is executed at the site where the transaction is submitted, and a set of other process, called cohorts, which execute on behalf of the transaction at the various sites that are accessed by the transaction).

20. Haritsa does not discloses dispatching the two-phase commit protocol operation to a second server thread if a second server thread is determined to be available. However, Lampson in view of Nageswaran discloses dispatching the two-phase commit protocol operation to a second server thread if a second server thread is determined to be available. Lampson discloses dispatching the second phase of the two phase commit protocol (the two phase commit protocol operation) to a second server thread (see fig. 4). Nageswaran teaches the concept of reusable threads and determining if they are available (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, but also to distribute the threads when needed by the application. It would also be obvious

to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

21. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit message to a subordinate (thread). Once the subordinate has the message it must commit. If the thread is a reusable thread, as described in Nageswaran, the system must look for a thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

22. Lampson in view of Nageswaran and in further view of Haritsa does not disclose processing the two-phase commit protocol operation in the primary thread if no second server thread is determined to be available and processing the remaining two-phase commit protocol operation in the first server thread. However, it would be obvious to one of ordinary skill in the art to appreciate that if there are no threads available, the primary thread should continue the 2 phase commit protocol process in a serial manner. If all the threads where busy, once the threads finished their procedures, they could continue to process the protocols that were not able to get processed. The advantages of

this process would be to make sure all the protocols were completed and not abandoned because there were no threads available.

23. In reference to claim 7, claim 7 depends upon claim 6. Claim 6 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792) and in further view of Haritsa (The PROMPT Real Time Commit Protocol, Feb. 2002).

24. Lampson in view of Nageswaran discloses dispatching the two-phase commit protocol operation to a second server thread (see Lampson fig. 2) including determining available serer threads in a server; and selecting an idle server thread to process the first two-phase commit protocol operation (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, but also to distribute the threads when needed by the application. It would also be obvious to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

25. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit message to a subordinate (thread). Once the subordinate has the message it must commit. If the thread is a reusable thread, as described in Nageswaran, the system must look for a

thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

26. In reference to claim 8, claim 8 depends upon claim7. Claim 7 has been obviated by Lampson et al. (5,335,343) in view of Nageswaran (5,991,792) and in further view of Haritsa (The PROMPT Real Time Commit Protocol, Feb. 2002).

27. Nageswaran discloses a thread pool manager determines the available server threads (see fig. 2, in which Nageswaran discloses a thread manager which controls the idle thread queue. See column 2, lines 40-50, in which Nageswaran discloses the thread manager maintain a dedicated thread ID with a particular client or transaction ID.) The thread manager manages the amount of idle reusable threads. It would be obvious to one of ordinary skill in the art to allow the thread manager to not only identify the idle reusable thread, available server threads, but can also distribute the threads when needed by the application. It would also be obvious to one of ordinary skill in the art to appreciate that the thread manager can select an idle thread for a 2 phase commit protocol operation because the thread manager is in control of the idle thread queue. The thread manager also controls the thread process. It would be able to determine if the server threads are available.

28. Once the first phase of the two phase commit protocol has completed, the second phase (the commit phase) can begin. The commit coordinator must send the commit

message to a subordinate (thread). Once the subordinate has the message it must commit. If the thread is a reusable thread, as described in Nageswaran, the system must look for a thread or be assigned a thread that is idle in order to carry out the second phase of the two phase commit protocol. The advantage of selecting an idle server thread to process the first two-phase commit protocol operation, is that the threads will be reusable. If we have reusable threads, then fewer resources are being wasted when the threads are not being used. Also, this means the 2 phase commit protocol is running in parallel in stead of serially and saving processing time.

29. In reference to claim 9, Lampson discloses each of the N two phase commit protocol operations is associated with a first phase of a two phase commit protocol. Lampson also discloses processing a second two-phase commit protocol operation by the first thread, the second two-phase commit protocol operation associated with a second resource and the first phase of the two phase commit protocol (see fig 4, 5, 6 and column 6, lines 45-50, in which Lampson discloses item sends "commit" message to all of its subordinates). Lampson discloses having multiple threads. Therefore, it would be obvious to one of ordinary skill in the art that there will be a second two phase commit protocol operation in another thread. Lampson is dispatching a phase 2 commit protocol operation of a two-phase commit protocol from the commit coordinator to the subordinates. One of ordinary skill in the art can appreciate that the subordinate could be considered a thread. The commit coordinator keeps track of the responses from the subordinates of the first phase (prepare to commit phase) of the 2 phase commit protocol. The commit coordinator is in charge for the first phase of the multiple subordinates that will carry out the second phase of the two phase commit protocol. Only if the

subordinate votes yes to commit during the first phase, does the second commit phase begin. Therefore, the commit coordinator has association with the subordinate. Also, the first phase effects the second phase. So, there is association between the first and the second phase of the 2 phase commit protocol. The advantage to the association between the first and second phase of the two phase commit protocol is, only if there is a vote of yes, can the protocol commit. Therefore the association is needed in order to go from phase 1 to phase 2 of the two-phase commit protocol.

30. In reference to claim 10, Lampson in view of Nageswaran does not disclose reporting results of the first and second two-phase commit protocol to a log. However, Haritsa discloses a 2 phase commit protocol where after the second phase of the 2 phase commit protocol, the results of the first and second 2 phase commit protocol are reported to a log (see pg. 162, section 3.1, third paragraph, in which Haritsa discloses each cohort, upon receiving the COMMIT message, moves to the committing state, force-writes a commit log record, and sends an ACK message to the master).

31. The motivation to combine the 2 phase commit protocol with reusable threads, as disclosed by Lampson in view of Nageswaran, with a commit log record is the commit log record will be able to keep records of what protocol has been committed and what has been aborted. This information can be valuable to the IT specialists caring for the system or running the two phase commit protocol.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jyoti D. Dave whose telephone number is 571-270-1470.

The examiner can normally be reached on 7:30 AM to 5 PM Mon-Fri, Alt Fri. Eastern Time.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Thu Nguyen can be reached on 571-272-6967. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Jyoti D Dave/ 12/20/2007
Examiner, Art Unit 4182

/Thu Nguyen/

Supervisory Patent Examiner, Art Unit 4182